REMARKS

Claims 1-9, 11, 13-23, 25-46, 48-54, 56-66, 70-81, and 83 are pending in this application. Claims 10, 12, 55, and 82 have been previously canceled. Claims 24, 47, and 67-69 are currently canceled. Applicant has amended claims 1-3, 32, 54, 70, 72, and 79 to address typographical and stylistic issues, and to more particularly point out and distinctly claim Applicant's invention. Support for these amendments can be found in the claims (*e.g.*, original claim 24), the figures (*e.g.*, Figures 5 and 17), and the text (*e.g.*, Applicant's Specification, page 11, line 20 – page 12, line 14; page 25, line 18 – page 16, line 6.) No new matter has been introduced by way of these amendments.

Overview

The Examiner has rejected claims 1-9, 11, 13-15, 30, 32-40, 50, 52-64, 66, 71-79, 81, and 83 under 35 U.S.C § 102(b) as being anticipated by Hansen et al. ("Hansen"), U.S. Patent No. 5,832,263. The Examiner has also rejected claims 1-4, 27-29, 32, 50-52, 54, 55, 67-69, 72, 79, and 83 under 35 U.S.C § 102(e) as being anticipated by White et al. ("White"), U.S. Patent No. 6,092,161. In addition, the Examiner has rejected claims 16-26, 31, 41-49, 65, 70, and 80 under 35 U.S.C § 103(a) as obvious over Hansen in view of Kobayashi et al. ("Kobayashi"), U.S. Patent No. 5,437,018.

Applicant respectfully traverses all these rejections for the reasons discussed in detail below with respect to both the original and amended claims as indicated.

Applicant notes the amendments provided are clarifying amendments and do not necessitate a new search, as these concepts from prior claims should have been searched by the Examiner already.

35 U.S.C § 102 and § 103 Rejections Based on Hansen and/or Kobayashi

The Examiner appears to be arguing that Hansen's described techniques, which include providing an illusion of modifiability for otherwise non-modifiable storage devices, and/or Kobayashi's described techniques, which include providing a semiconductor auxiliary storage device that permits modification to programs and/or data stored in read-only storage on the device, somehow teach, suggest, or motivate Applicant's claimed techniques for "providing a

storage redirection driver that protects the storage devices of a computer system from alteration." (Specification, p.1, lines 11-12.)

Applicant respectfully disagrees for several reasons.

First, Applicant's claims recite several aspects that are nowhere taught, suggested, or motivated by Hansen or Kobayashi. Each of independent claims 1-3, 32, 54, 72, and 79, as amended, recite processing requests to modify a location in unprotected space by *initiating modification of the location in the unprotected space without redirection.* Specifically, claim 1 recites, "when the request is to modify a location in the unprotected space, initiating modification of the location in the unprotected space without redirection." Claim 2 recites, "when the request is to modify a location in the unprotected space, initiate modification of the location in the unprotected space without redirection." Claim 3 (and claims 4-9, 11, 13-23 and 25-31, by virtue of their dependencies) recites, "when the request is to modify a location in the unprotected space, initiating modification of the location in the unprotected space without redirection." Claim 32 (and claims 33-46 and 48-53, by virtue of their dependencies) recites, "when the request is to modify a location in the unprotected space, initiating modification of the location in the unprotected space without redirection." Claim 54 (and claims 56-66 and 70-71, by virtue of their dependencies) recites, "when the request is to modify a location in the unprotected space, initiate modification of the location in the unprotected space without redirection." Claim 72 (and claims 73-78, by virtue of their dependencies) recites, "when the request is to modify a location in the unprotected space, initiating modification of the location in the unprotected space without redirection." Claim 79 (and claims 80, 81 and 83, by virtue of their dependencies) recites, "when the request is to modify a location in the unprotected space, initiate modification of the location in the unprotected space without redirection."

Hansen does not teach, suggest, or motivate processing requests to "modify a location in the *unprotected space* [by initiating] modification of the location in unprotected space without redirection" as recited (emphasis added). In particular, Hansen appears to only process requests that are directed to a read-only storage, by mapping them to a modifiable storage area. For example, Hansen states, "The method and system intercepts read and write requests targetted [sic] at the read-only storage and remaps them when appropriate to a modifiable storage area." (Hansen, Abstract.) Applicant has reviewed Hansen in detail and cannot find any teaching or

suggestion of processing requests to modify a location in unprotected space or even to differentiate between requests to modify a location in the unprotected space and requests to modify a location in the protected space, in the manner recited by Applicant's claims.

Nor does Kobayashi teach, suggest, or motivate processing requests to modify a location in unprotected space in the manner recited by Applicant's claims. The Examiner asserts that Kobayashi describes handling accesses to an unprotected space of the storage device at column 2, line 51 to column 3 line 11. (Office Action dated March 13, 2007, hereinafter "Office Action," page 25, with respect to claims 24, 47, 67, and 69.) In particular, the Examiner appears to be equating Kobayashi's RAM area with Applicant's unprotected space. (*Id.*) However, Kobayashi's RAM area cannot be equated with Applicant's unprotected space because every access to the RAM area 58 in Kobayashi is actually the result of a *redirected* access to a first ROM area 56. (Kobayashi, Figs 10A-10C and 12, column 9, line 43 – column 10, line 61, hereinafter in column#:line# format.) In particular, Figure 12 describes a process of converting a logical address into a physical address of the semiconductor storage device. If the logical address is smaller than the first ROM area 56, then it is translated to a physical address of the RAM area 58. (Kobayashi, 10:48-57.) In this case, any access to the RAM area 58 is occurring *with* redirection, and therefore Kobayashi does not teach or suggest modification of unprotected space *without* redirection. On the other hand, if the logical address is larger than the first ROM 56 area, it "designates a point in the second ROM area 57." (Kobayashi, 10:60-61.) Accordingly, Kobayashi does not describe or contemplate any accesses to the RAM area 58 *without redirection.* Thus, Kobayashi does not anticipate claims 1-9, 11, 13-23, 25-46, 48-54, 55-56, 70-81, and 83.

Furthermore, it is not the case that Kobayashi's second ROM area 57 can be equated with Applicant's unprotected space. As is clear from the language of the Applicant's claims quoted above, requests to *modify* the unprotected space are processed by initiating *modification* of the unprotected space. It is well known that ROMs, such as Kobayashi's ROM area 57, cannot be modified. ("ROM: Any semiconductor storage serving as a memory that contains instructions or data than can be read but not modified." *Microsoft Computer Dictionary*, 3rd ed., 1997.) So even if a modification request to the second ROM area 57 were received, such a request could never result in a modification, as recited by Applicant's claims.

Accordingly, neither Hansen nor Kobayashi teach, suggest, or motivate processing requests "to modify a location in the unprotected space" as recited by Applicant's claims.

In addition, Hansen combined with Kobayashi does not teach, suggest, or motivate the above recited aspects. The Examiner has combined these references to reject several of the dependent claims. Applicant notes that, even if a combination of Hansen and Kobayashi can be considered to teach all of the aspects of Applicant's claims, which it cannot, one skilled in the art would not have been motivated to combine those references. First, Kobayashi and Hansen are non-analogous art in that Kobayashi presents a firmware/hardware-oriented approach, whereas Hansen presents a software-oriented approach. In particular, Kobayashi utilizes instructions in a BIOS-resident ROM to convert accesses to magnetic storage (*e.g.*, floppy drives) instead into accesses to semiconductor auxiliary storage. (Kobayashi, 5:62-64.) That Kobayashi is a BIOS-based solution is made clear in the description of Figure 5, which is "a flow chart of the read write processing of BIOS 43 for software interrupt INT1Bh from DOS 42." (Kobayashi, 6:7-9.) In short, Kobayashi utilizes a hardware/firmware-based approach to provide transparent access to auxiliary semiconductor storage by unmodified operating system code. (Kobayashi, Abstract and 6:2-6.) Hansen, on the other hand, is a software-based approach, utilizing custom device drivers, software modules, or plugins, thereby modifying the existing code. (Hansen, 4:47-48 and 6:34.) Because Hansen and Kobayashi present two very different approaches to providing the illusion of modifiability of un-modifiable storage, one skilled in the art would be unlikely to combine the references to achieve the protected and unprotected modification requests recited in the claims.

Second, applying Hansen's teachings to Kobayashi would render Kobayashi inoperable for its intended purpose. A central purpose of Kobayashi is to convert "access requests for the semiconductor auxiliary storage ... without requiring any modification of existing programs, such as, application software and disk operating systems." (Kobayashi, Abstract.) Kobayashi accomplishes this purpose by way of a firmware/hardware-based approach that uses a modified BIOS coupled with an auxiliary storage device, as discussed above. The BIOS-based solution allows an unmodified operating system and applications to run on the auxiliary storage device. (Kobayashi, 6:2-9.) To apply the software-based teachings of Hansen (*e.g.*, to include a software module or device driver) to Kobayashi would mean that Kobayashi

would no longer be able to provide support for *unmodified* operating systems or application programs, because such operating systems and programs would necessarily be modified by Hansen's drivers, software modules, or plugins. Accordingly, the software-oriented teachings of Hansen would render Kobayashi inoperable for Kobayashi's intended purpose of providing support for unmodified programs and operating systems.

Third, one skilled in the art would not be motivated to apply the teachings of Kobayashi to those of Hansen, because Hansen specifically teaches away from using techniques such as those described by Kobayashi. In particular, the background section of Hansen specifically discusses, and points out numerous disadvantages with, techniques such as those described by Kobayashi. In particular, it notes that by supporting "modification" of non-modifiable stores (*e.g.*, Kobayashi's ROM 56) by storing modifications in a RAM (*e.g.*, Kobayashi's RAM 58), such modifications may not be retained without physically replacing the non-modifiable store (*e.g.*, replacing the ROM). (Hansen, 1:40-47.) In addition, Hansen points out that using a RAM to store changes to a non-modifiable store is inherently limited by the size of the RAM used for storing modifications. (Hansen, 1:56.) Accordingly, one skilled in the art, upon reading Hansen, would not be motivated to turn to the RAM-based approach of Kobayashi.

Thus, Hansen and Kobayashi combined do not render obvious any of the claims 1-9, 11, 13-23, 25-46, 48-54, 56-66, 70-81 and 83.

35 U.S.C § 102 Rejections Based on White

The Examiner appears to be arguing that White's described techniques, which provide a hardware supervisor for virus protection, somehow teach, suggest, or motivate Applicant's claimed techniques for "providing a storage redirection driver that protects the storage devices of a computer system from alteration." (Specification, p.1, lines 11-12.)

Applicant respectfully disagrees. Enclosed is a Rule 132 Declaration of Dr. Randy Keith Lomnes. Dr. Lomnes is the inventor of the above-identified patent application, a co-founder of Hyper Technologies, Inc. (the current assignee of the above-identified patent application), and has extensive experience in both hardware and software design and implementation. (Rule 132 Declaration of Dr. Lomnes, hereinafter "Lomnes Declaration," ¶¶ 1 and 3.) Dr. White discusses in detail the approach taken by White, and how that approach differs

in various ways from several aspects recited by the claims. (Lomnes Declaration, ¶¶ 4-10.) In particular, as will be discussed further below, Dr. Lomnes explains in detail why White does not teach, suggest, or motivate several aspects recited by independent claims 1-3, 32, 54, 72, and 79, both before and after amendment.

The Examiner asserts that White teaches a "software redirection driver" as recited by Applicant's claims, because, it is argued, "firmware is a special type of software ... stored in read-only memory." (Office Action dated November 30, 2006, p. 5.) Applicant respectfully disagrees, as independent claims 1-3, 32, 54, 72, and 79, prior to amendment, each recite *installing* or *loading* code "during power-up initialization" or "from a powered-down state." As Dr. Lomnes explains, White operates by executing instructions from a ROM. (Lomnes Declaration, ¶ 4.) Dr. Lomnes further explains that such ROMs are typically burned by a manufacturer, and can be modified only by replacing the ROM chips themselves. (Lomnes Declaration, ¶ 10.)[1] In addition, Dr. Lomnes notes that White does not disclose any circuitry that would be needed to modify (*e.g.*, program) the contents of the ROM once the card is in use. (Lomnes Declaration, ¶ 10.) Accordingly, White's ROM-resident instructions cannot be said to be installed or loaded during power-up initialization, as they are permanently resident therein. Nonetheless, Applicant has amended independent claims 1-3, 32, 54, 72, and 79 to more particularly point out and distinctly claim Applicant's invention, as discussed in more detail below.

Applicant's amended claims recite several aspects that are nowhere taught, suggested, or motivated by White. Each of independent claims 1-3, 32, 54, 72, and 79, as amended, recite a *software redirection driver* that performs data protection functions and that is installed or loaded into *a volatile memory* of a computer system during system startup. Specifically, claim 1 recites, "loading the software redirection driver code into a volatile memory of the computer system during the starting of the computer system." Claim 2 recites, "a volatile memory; ...; a software redirection driver, installed in the volatile memory during power-up initialization." Claim 3 (and its respective dependent claims) recites, "loading a software redirection driver into a volatile memory of the computer system during power-up initialization."

[1] *See also*, Microsoft Computer Dictionary 3rd ed., 1997 ("ROM: A semiconductor circuit into which code or data is permanently installed by the manufacturing process.")

Claim 32 (and its respective dependent claims) recites, "loading a software redirection driver into a volatile memory of the computer system during power-up initialization." Claim 54 (and its respective dependent claims) recites, "a software redirection driver, loaded into a volatile memory of the computer system when the system is booted from a powered-down state." Claim 72 (and its respective dependent claims) recites, "loading a software redirection driver into a volatile memory of the computer system during power-up initialization." Claim 79 (and its respective dependent claims) recites, "a software redirection driver, installed in a volatile memory of the computer system upon power-up initialization."

White, in contrast, does not teach, suggest, or motivate software redirection drivers installed in volatile memory upon power-up initialization, as recited by Applicant's claims, let alone redirection drivers or other software components that perform the remaining aspects of Applicant's independent claims. The Examiner asserts that White discloses the recited aspects at column 1, lines 36-39. (Office Action, page 12, with respect to claim 1.) The cited passage describes a "supervising means causing a reset to be required of the computer system" and thereby "causing memory to be cleared and an operating system to be loaded." As discussed below by Dr. Lomnes, the cited passage in particular, and the White reference in general, do not teach, suggest, or motivate the recited software redirection driver for a number of reasons.

As discussed in Applicant's Specification, drivers may be utilized by an operating system to process accesses to storage devices and other hardware. (*See*, Applicant's Specification, Figure 2 and page 8, line 13 – page 9, line 3 (describing device drivers); Figure 9 and page 15 line 1 – 23 (describing a layered Input/Output driver architecture of a typical operating system).) A software redirection driver is a type of operating system driver. (Applicant's Specification, page 8 lines 13-18.) As discussed by Dr. Lomnes, the functions of White are not performed by a *software redirection driver*. Dr. Lomnes states that White does not describe a software driver-based approach to intercepting disk accesses. (Lomnes Declaration, ¶ 7.) Dr. Lomnes indicates that the functions of White are instead performed by a supervisor computer processor executing on a peripheral card that is separate from an operating system executing on a personal computer. (Lomnes Declaration, ¶¶ 4-6.) As further discussed by Dr. Lomnes, the peripheral card of White is not the same thing as a software redirection driver. (Lomnes Declaration, ¶¶ 5 and 7.) In particular, the peripheral card of White is separate from,

and does not interact with, the operating system of the personal computer. (Lomnes Declaration, ¶ 5.) Further, the peripheral card of White is connected by ribbon cables to the personal computer and to the disk drive, and thus is physically separated. (Lomnes Declaration, ¶ 5.) Accordingly, White does not teach, suggest, or motivate a software redirection driver, as recited by Applicant's claims.

In addition, as discussed by Dr. Lomnes, White does not disclose any software component that is loaded into memory during system power-up or initialization *and* that performs the functions/operations recited by Applicant's amended claims. Applicant's amended claims continue to recite loading a driver during "power-up initialization" or "when a system is booted from a powered-down state." As noted above, with respect to Applicant's claims prior to amendment, and as further discussed by Dr. Lomnes, ROM-resident instructions (as used by White's ROM-based supervisor) are not loaded into the ROM at system startup or initialization, but rather during manufacture. (Lomnes Declaration, ¶¶ 4, 10.) Accordingly, White does not teach, suggest, or motivate loading a software component at system startup or initialization, as recited by Applicant's amended claims.

Furthermore, as discussed by Dr. Lomnes, White's supervisor does not include a software redirection driver that resides in *volatile memory*. Dr. Lomnes indicates that White utilizes firmware that includes supervisor instructions stored in a read-only memory ("ROM") and *not* a volatile memory. (Lomnes Declaration ¶¶ 4 and 9.) Dr. Lomnes notes that the types of ROMs used in White are typically burned by a manufacturer. (Lomnes Declaration, ¶ 10.) As such, the ROM used in White is not configured to be changed after its manufacture, and accordingly cannot be considered volatile memory.[2] In addition, as Dr. Lomnes further discusses, White would not be operable for its intended purpose of protecting a disk from malicious software if it were modified to operate using a user-programmable ROM or with a volatile memory such as a RAM. (Lomnes Declaration, ¶¶ 9-10.) Thus, White does not teach, suggest, or motivate the use of a *volatile memory* as recited by Applicant's claims.

[2] *Compare*, Microsoft Computer Dictionary, 3rd ed., 1997 ("Volatile memory: Memory, such as RAM, that loses its data when the power is shut off") *Id.*, ("Nonvolatile Memory: A storage system that does not lose data when power is removed from it. Intended to refer to core memory, ROM, EPROM, flash memory, ...")

Furthermore, a combination of White with any of the other cited references would not render obvious Applicant's claims. In particular, combining White with the teachings of Kobayashi does not teach, suggest, or motivate a software redirection driver. As noted above, Kobayashi also describes a firmware/hardware-oriented approach, in its reliance upon a ROM-resident BIOS that translates memory accesses for an semiconductor auxiliary storage device. Because neither White nor Kobayashi teach a software redirection driver, a combination of the two references cannot, by definition, teach all aspects of Applicant's independent claims.

In addition, a combination of White and Hansen does not teach, suggest, or motivate a software redirection driver that functions as recited by Applicant's claims. Even if one could somehow argue that a combination of White and Hansen could teach all aspects of Applicant's claims (which they cannot), one skilled in the art would not have been motivated to combine these references because such a combination would render White inoperable. Thus, White and Hansen cannot be legitimately combined to teach, suggest, or motivate Applicant's claims. In particular, if the supervisor of White were implemented as a software component as discussed in Hansen, White would be inoperable for its intended purpose of virus protection. White itself states that a separate supervisor processor that executes ROM-based instructions is central to its purpose: "a virus can never interfere with the Supervisor microprocessor 216 since it is *only* able to fetch executable code from its own ROM 213." (White, 10:16-18, emphasis added.) White further states that Figures 7 and 8 "clearly [show] that a virus can never interfere with the Supervisor microprocessor 314 since it is *only* able to fetch executable code from its own ROM 326." (White, 11:27-29, emphasis added.) In short, White can only perform its function of protecting the computing system from viruses by virtue of the fact that it is implemented as an external hardware component that monitors and possibly restricts read/write requests to a storage medium.

Also, Dr. Lomnes indicates that the purpose of White is to protect a disk from malicious software. (Lomnes Declaration, ¶ 4.) Dr. Lomnes further states that the purpose of White would be defeated by modifying the hardware-oriented approach of White to operate as a software driver, because hackers could tamper with the software in order to gain access to the disk accesses. (Lomnes Declaration, ¶¶ 8-9.) White can only perform its function of protecting the computing system from viruses by virtue of the fact that it is implemented as an external

hardware component that controls all communication between a personal computer and a disk drive. (Lomnes Declaration, ¶ 5.) Accordingly, one skilled in the art would not be motivated to implement the supervisor of White as a software driver as recited by Applicant's claims.

Conclusion

Thus, because at least one aspect of independent claims 1-3, 32, 54, 72, and 79 is not taught, suggested or motivated by Hansen, Kobayashi, or White, claims 1-3, 32, 54, 72, and 79 are not anticipated by, or obvious in view of any motivated combination of, Hansen, Kobayashi, and White. Similarly, because dependent claims 4-9, 11, 13-23, 25-31, 33-46, 48-53, 56-66, 70-71, 73-78, 80-81, and 83 incorporate the respective aspects of claims 1-3, 32, 54, 72, and 79 by virtue of their dependencies, the dependent claims also are not anticipated by or rendered obvious in view of Hansen, White, or Kobayashi, alone or in any motivated combination, for at least the reasons set forth above.

The Examiner has also rejected many of the dependent claims for different reasons. Applicant traverses these rejections and notes dependent claims 4-9, 11, 13-23, 25-31, 33-46, 48-53, 56-66, 70-71, 73-78, 80-81, and 83 are also not taught, suggested, or motivated by Hansen, Kobayashi, or White for a variety of additional reasons. For example, with respect to the rejection of claim 5, neither Hansen nor White teach, suggest, or motivate wherein "the driver is inserted into a driver hierarchy." Also, with respect to the rejection of claim 64, neither Hansen nor White teach, suggest, or motivate that the driver "refers to redirected space using multiple data addressing abstractions." In the interests of expediting prosecution, such arguments are not addressed in more detail herein. Accordingly, Applicant reserves the right to further traverse these rejections if necessary.

In the event the Examiner disagrees with applicant or finds minor informalities, Applicant respectfully requests a telephone interview to discuss the Examiner's issues and to expeditiously resolve prosecution of this application. Accompanying this Amendment is an Applicant Initiated Interview Request Form in the event the Examiner does not agree that the claims are allowable over the cited references. Applicant's representative can be contacted at (206) 622-4900.

In closing, Applicant respectfully submits that all of the pending claims are allowable and respectfully requests the Examiner to enter these amendments and to reconsider this application and its timely allowance. The Director is authorized to charge any additional fees due by way of this Amendment, or credit any overpayment, to our Deposit Account No. 19-1090. Again, Applicant's representative thanks the Examiner for his prompt and courteous attention.

Respectfully submitted,

SEED Intellectual Property Law Group PLLC

Ellen M. Bierman

Ellen M. Bierman
Registration No. 38,079

EMB:asl

Enclosures:

Rule 1.132 Declaration of Randy Keith Lomnes, Ph.D.
Applicant Initiated Interview Request Form

701 Fifth Avenue, Suite 5400
Seattle, Washington 98104
Phone: (206) 622-4900
Fax: (206) 682-6031
977362_1.DOC